CD for DBs
Database Deployment Strategies

Chris Fulton, Global Technical Account Manager, Electric Cloud
Evolution of Database Deployments

Manual Prod  Manual DB  Auto DBI  Auto DB++  DB Anywhere
Why is the Database Important

• Contains logic and critical business data

• The “brains” of an application

• Making changes to a database is much like doing brain surgery
Why is it complicated

• More than one application...

• Multiple databases per server...

• Most critical data...
  • If you lose data from a database, you lose critical customer data

• Ouch. Schema Changes
Things to consider
Testing

• What applications does this deployment affect

• What testing should be done (before/during/after)

• What do you do if testing breaks
  o Automated Rollback?
  o Manual Intervention?
  o Who decides
Rollback

• At what point did it fail
  ▪ Did I corrupt tables
  ▪ Do I have partially updated data
  ▪ Is it partially deployed

• Was the system running when updating
  ▪ Do I have some records updated/changed since start
Automation

• Who owns the automation?
• How does the automation determine success/fail?
• What if I do fail?
• What tool to use?
• Supported features
  ▪ Automatic Rollback?
  ▪ Rolling Deployments?
Distributed Development / Infrastructure

• How do developers merge code
• How do you integrate developers work
• How do you handle distributed systems across the world
Methods
SQL Scripts / Rollback Scripts

```sql
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';

DROP SCHEMA IF EXISTS consult;
CREATE SCHEMA consult;
USE consult;

CREATE TABLE address (  
    address_id INTEGER NOT NULL AUTO_INCREMENT,  
    line1 VARCHAR(50) NOT NULL,  
    line2 VARCHAR(50) NULL,  
    city VARCHAR(50) NOT NULL,  
    region VARCHAR(50) NOT NULL,  
    country VARCHAR(50) NOT NULL,  
    postal_code VARCHAR(50) NOT NULL,  
    CONSTRAINT address_pk PRIMARY KEY ( address_id )
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE consultant_status (  
    status_id CHAR NOT NULL,  
    description VARCHAR(50) NOT NULL,  
    CONSTRAINT consultant_status_pk PRIMARY KEY ( status_id )
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```
SQL Scripts / Rollback Scripts - Advantages

• Fast deployments
• Just run a SQL script!
• Minimal or no downtime

WHEN IT WORKS
SQL Scripts / Rollback Scripts - Disadvantages

• Requires developer to never make mistakes
• Requires extensive testing
  - How to TRULY test rollback
• Error prone
• How do you merge multiple developers work
SQL Scripts / Rollback Scripts - Who should use

- Small Databases
- Small development teams
- Simple changes
- Legacy Databases
  - If no other choice
Backup at Deploy
Backup at Deploy - Advantages

• Guarantees you are 100% back to previous good state

• “Safest” technically speaking
Backup at Deploy - Disadvantages

• Every application that depends on the database is 100% down during deploy

• Slower deployments
Backup at Deploy - Who should use

• Applications that can withstand being down

• Smaller databases that backup quickly

• Applications that do not share databases
DACPAC

• Microsoft’s solution to help solve Database Deployment issues
• Profile
  ▪ How do I run the DACPAC
• DACPAC
  ▪ What changes did I make

• Looks at what is on the database currently and compares to changes in DACPAC then applies changes
DACPAC - Advantages

• Creates easier rollbacks

• Deploy the previous DACPAC
  o ElectricFlow™ lets you do this Automatically!
  o Works well with multiple developers

• Use VisualStudio
DACPAC - Disadvantages

- Requires well written profile
  - Poorly written can cause data loss
- Microsoft only solution
  - Not for Oracle or other DB’s
- Relatively new
DACPAC - Who should use

• Microsoft only shops that are on a newer version of SqlServer

• All who use a version of SqlServer that supports them
Other DB Deployment tools

- Datical
- DBMaestro
- Redgate
- Many Many more
Strategies
Just do it - Deploy all at once

• Deploy all at once to all machines

• Run individual tests after deployment

• Rollback all at once
Just do it - Deploy all at once

• Advantages
  ▪ Super fast deployments
  ▪ On Success, things are already out there

• Disadvantages
  ▪ Dangerous if you have failures
  ▪ Could bring down entire system
  ▪ Not a lot of time to test

NOT RECOMMENDED
Partial deploy (Canary)

Deploy Database Cluster

Deploy Application Cluster
Partial Deploy - Advantages

• Allows you to test on a subset of users
• On failure, you do not bring down everyone. Simply re-point the users on the new deploy cluster back to the original cluster
• Allows you to give a “sneak peek” to a subset of users
Partial Deploy - Disadvantages

• Can be complex to maintain
• Requires more infrastructure to maintain more clusters
• Often requires application changes
• Requires synchronizing databases
Rolling deployment

Database Cluster

Application Cluster

Test
Rolling deployment - Advantages

• No clone of infrastructure required
• Allows you to do some testing before rolling out to all machines
  ○ If something breaks you can simply remove the machines you deployed to from the cluster (little to no downtime)
• If you already have a cluster for your DB and Application, little or no changes are required
• Natively built into ElectricFlow™
Rolling Deployment - Disadvantages

• Version mismatch could cause issues depending on the application and architecture

• Requires you to have clustered environment for your DB and Application
  ▪ (This is a best practice!!!)

• Could cause a complex overhead in scripts, unless using a tool like ElectricFlow™
Additive Strategy

• Add columns / rows and do not change existing ones
  • Copy existing columns to new column, then make changes on new column
• Deploy Database ahead of time
  • After Database is deployed, do testing on database while live database still uses existing columns
• After changes are validated, cleanup old columns / rows
Additive Strategy - Advantages

• Never mess with existing columns
• Less room for production errors
• Allows testing with “production” data
Additive Strategy - Disadvantages

• Can cause databases to be larger (especially if you are not rigorous at cleaning up)
• Can cause errors if conflicts between added columns and original columns
Take Aways
Things to remember

• Don’t treat the database as the stepchild
  o Consider Databases from the start

• Different solutions fit different situations

• Consider failure when coming up with a deployment strategy

• Follow best practices
  o Backups, Clones, testing failure, etc

• Version the database metadata along with the process
Talk to me

Email: cfulton@electric-cloud.com
LinkedIn: https://www.linkedin.com/in/chrisfulton
Company: http://www.electric-cloud.com
Try us out: http://electric-cloud.com/electricflow/
Thank you!